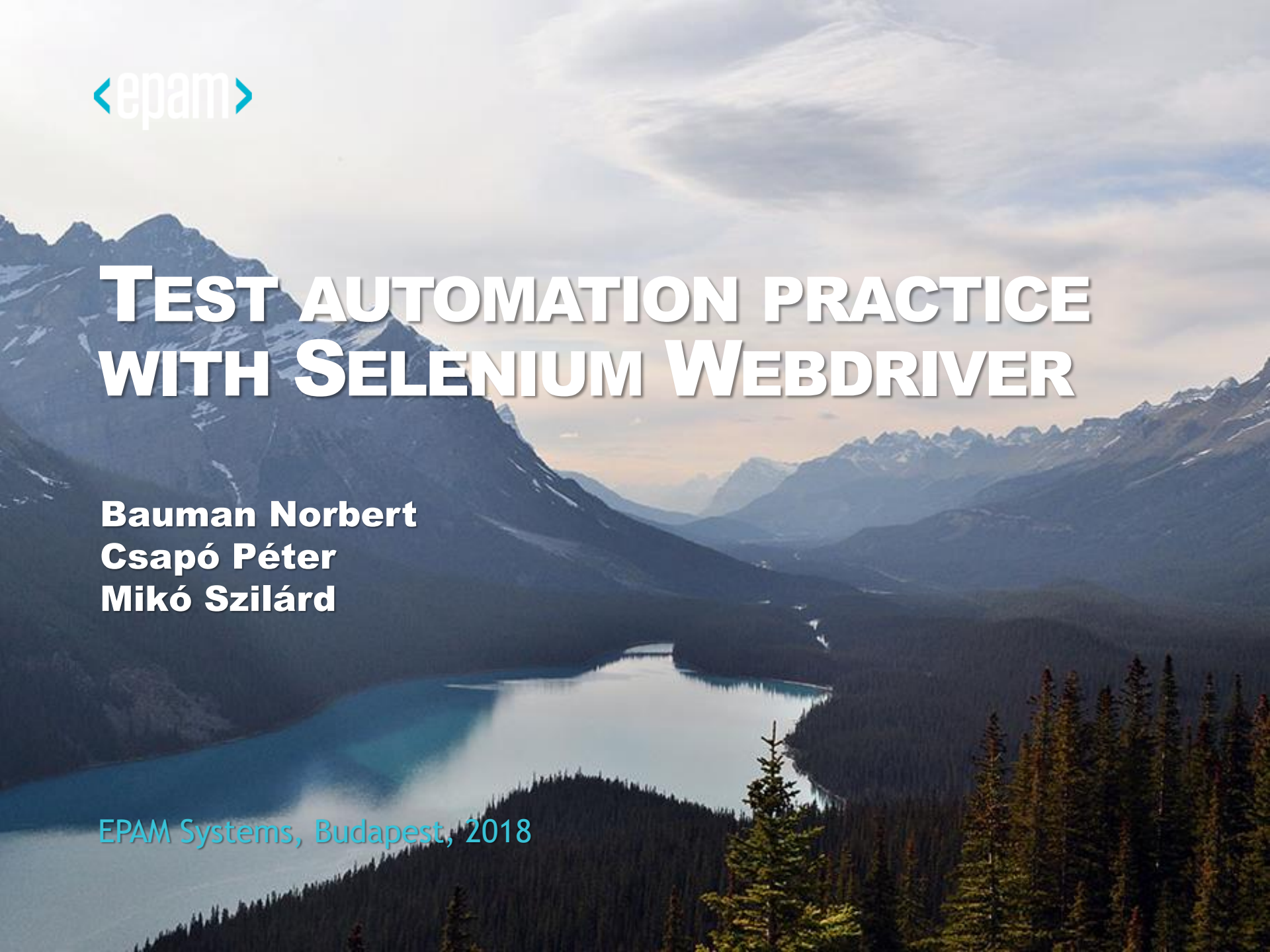

epam
Test automation practice with Selenium Webdriver
Bauman Norbert
Csapó Péter
Mikó Szilárd
EPAM Systems, Budapest, 2018

Agenda

1. Automated Web Testing in general
2. Record and Replay
3. Introduction to Selenium
4. Environment setup
5. Assertions
6. Navigations

CHAPTER 1

AUTOMATED WEB TESTING IN GENERAL

Automated (web) testing

- Using a software tool to run **repeatable** tests against the application to be tested
- Neccessarry for regression testing
- Simulate end user behavior as much as possible
  - Record and Playback
  - Language -> Driver -> Browser
- Plays nicely with TDD, BDD software development process

Why/When automate?

- Frequent regression testing
- Rapid feedback to developers
- Support for Agile
- Disciplined documentation of test cases
- Finding defects missed by manual testing

When NOT to automate?

- if the application's user interface will change considerably in the near future, then any automation might need to be rewritten anyway
- automation for images, texts, ads, overall outline of page might be not be worthwhile
- sometimes there is simply not enough time to build test automation.
- For the short term, manual testing may be more effective

CHAPTER 2

RECORD AND REPLAY

Record and Replay

- Record exact user interactions previously performed by the tester
- Easy to create test cases
- Ineffective repetition
- Limited options
- Limited verification capabilities
- e.g. Selenium IDE, iMacros, QTP, TestComplete

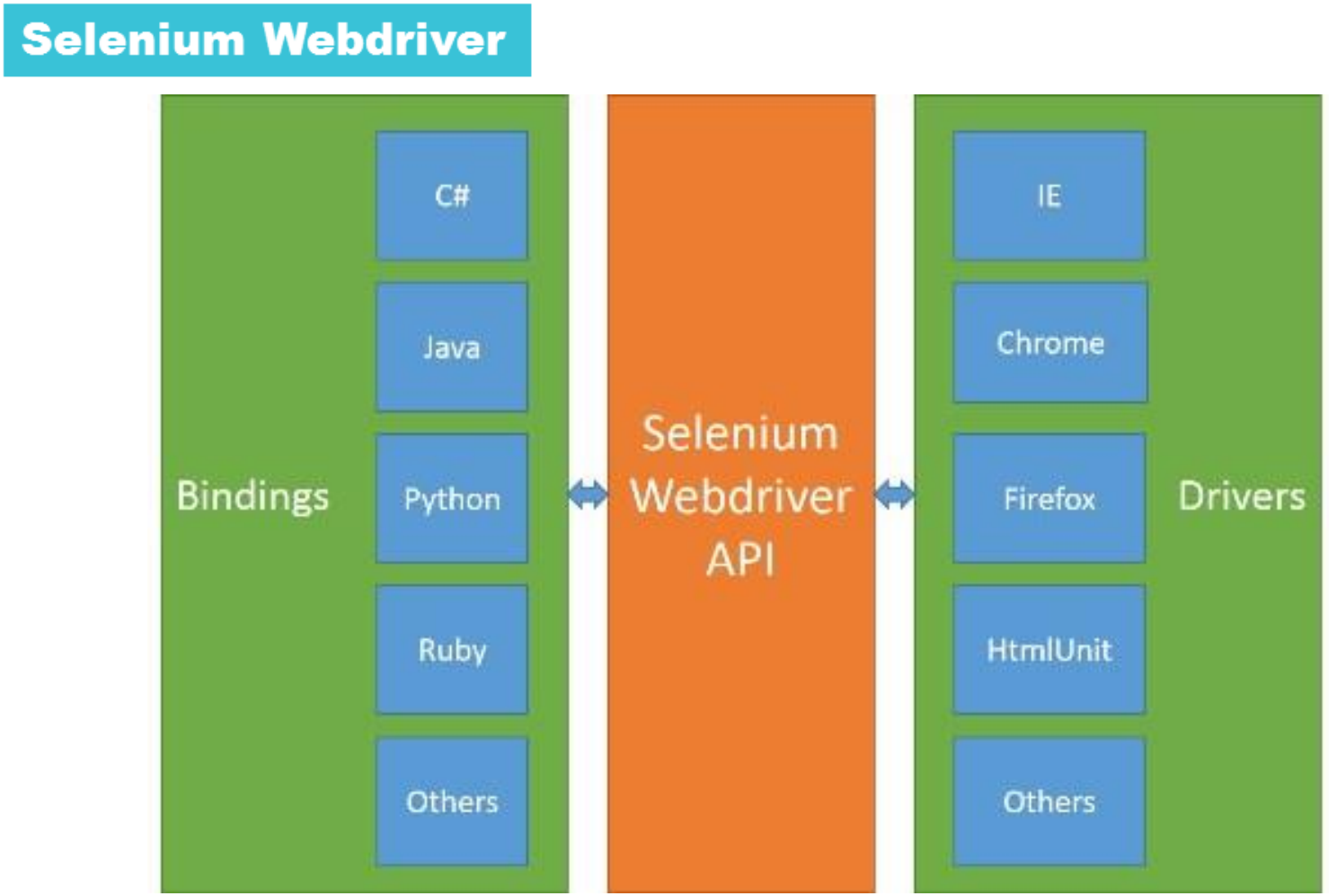
CHAPTER 3

SELENIUM WEBDRIVER

Script->Driver->Browser

- **Selenium Webdriver**
- Selenium Grid
- Watir
- Capybara
- PhantomJS

Script->Driver->Browser structure



Browsers

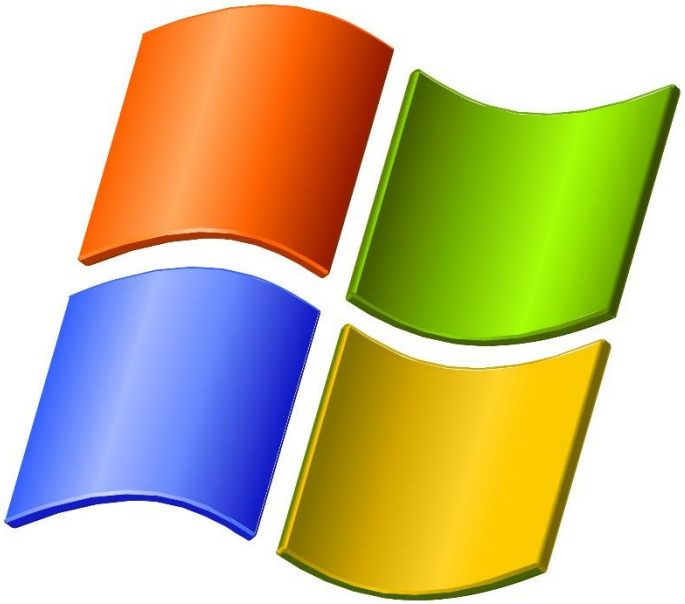
HtmlUnit

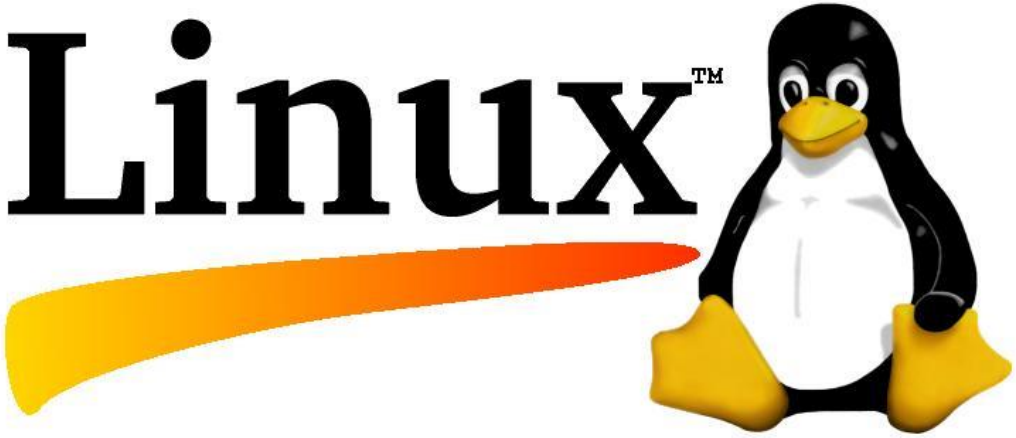
GUI-Less browser for Java programs

Challenges – Browser differences

- Major functionality is consistent but there can be subtle differences in:
  - Browser size
  - Screenshots
  - Element visibility
  - Performance
  - Native events
  - Cookie management
- IE is the worst offender

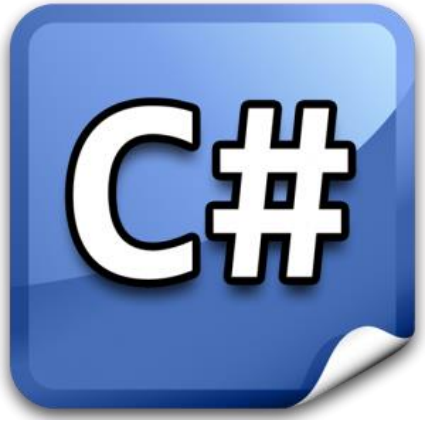
Supported OS





iOS

Supported Programming Languages



Ruby
A Programmer's Best Friend

JavaScript

Chapter 4

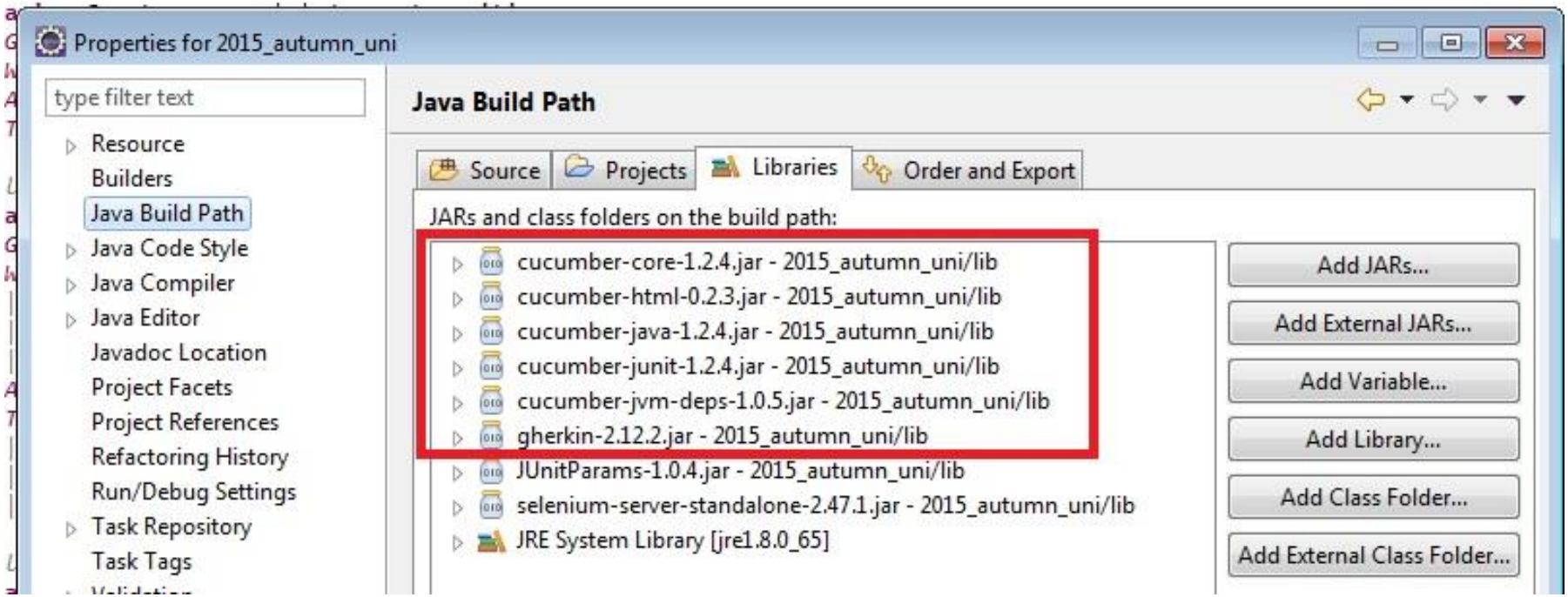
ENVIRONMENT SETUP

Eclipse setup

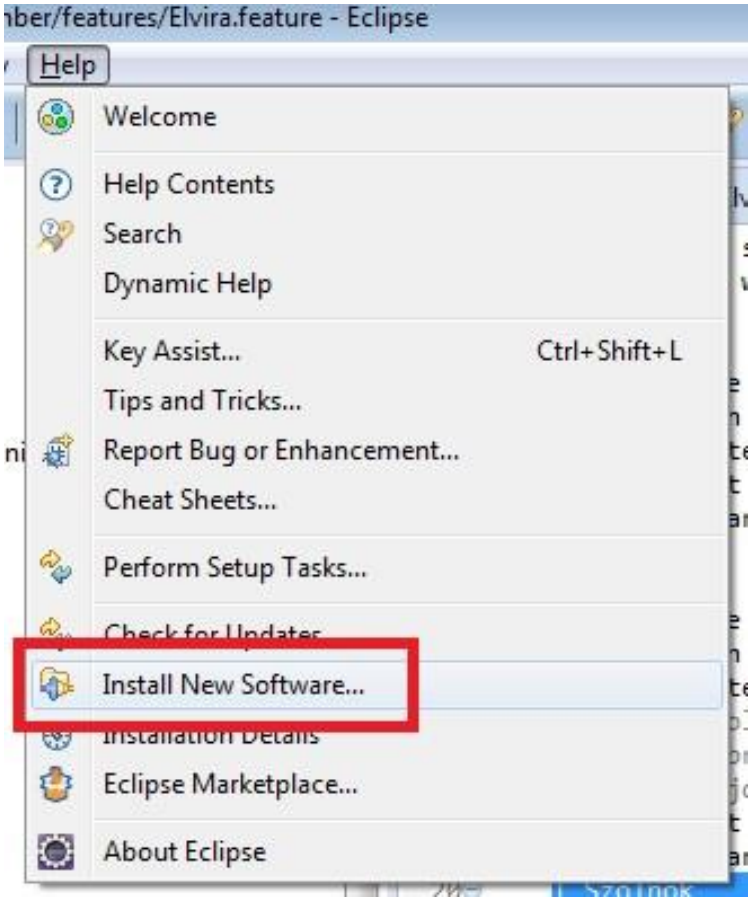
- Download eclipse
- Deploy eclipse: use administrator mode
- Open project:
    - Right click on Package Explorer -> Import... -> Existing Project
- Help -> Eclipse marketplace -> Natural 0.7.6
- Help -> Install new software -> work with <cucumber update site>
- Window -> Preferences -> General -> Workspace -> change character encode to UTF-8
- Start chrome driver in drivers folder (non administrator mode)
- Run test

Cucumber in Eclipse

Add JARs to your project



Cucumber in Eclipse



Help -> Install New Software

Work with: Cucumber

CHAPTER 5

ASSERTION

Assertion

Fail method

```
fail(error_message)
```

Conditional assert

```
assertTrue(error_message, boolean_condition)
```

Equality assert

```
assertEquals(error_message, expected, actual)
```

Provide meaningful messages in assertions!

Assertion

Identity assert

```
assertSame(error_message, expected_object, actual_object)
```

Custom assert

```
assertThat(actual_object, Matcher<object> matcher)
```

String assert

```
assertThat("myString", containsString("ring"))
```

Click here for more JUnit assertions

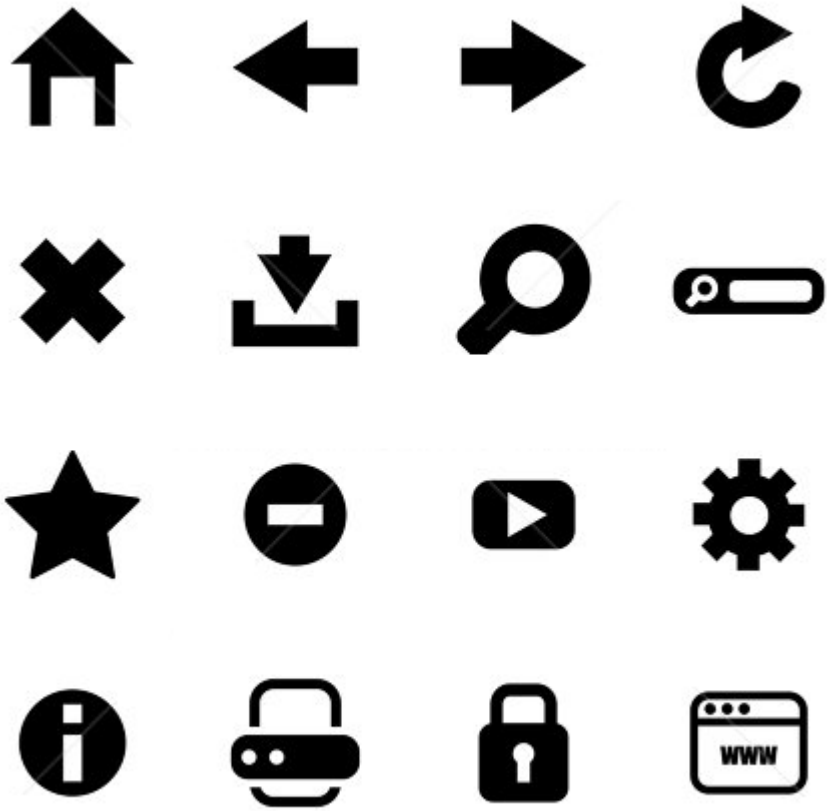
Benefits of assertThat

CHAPTER 6

NAVIGATION

What can you do with a browser?

- navigate to url
- get title, get url
- take screenshot
- refresh page,
- close browser/tab
- send keys
- execute script



Navigation

Loading a web page in current browser window

- driver.get(java.lang.String)
- driver.navigate().to(java.lang.String)
- driver.navigate().to(java.net.URL)

Navigation

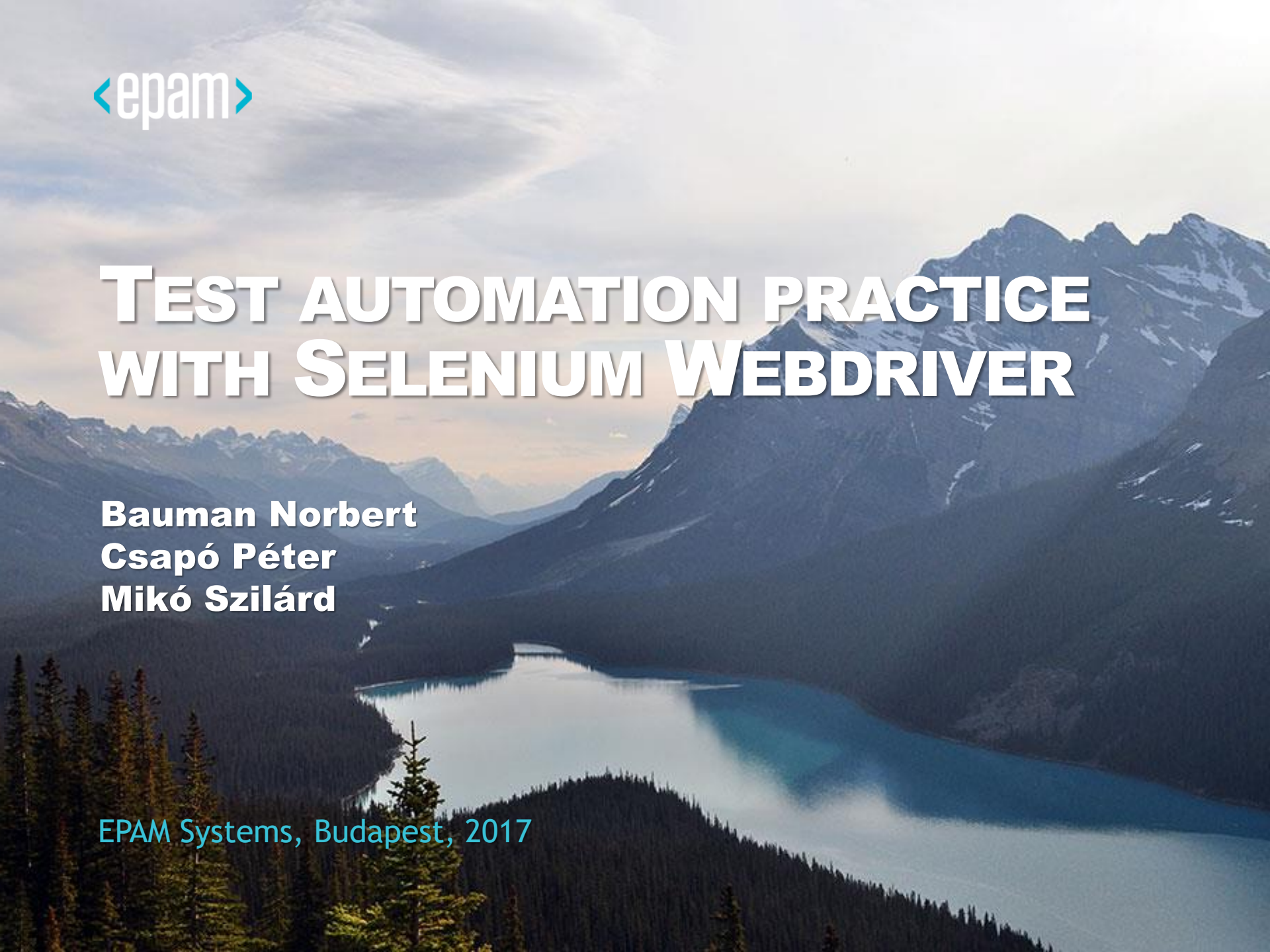
Move back & forward

- driver.navigate().back()
- driver.navigate().forward()

Refresh page

- driver.navigate().refresh()

Questions



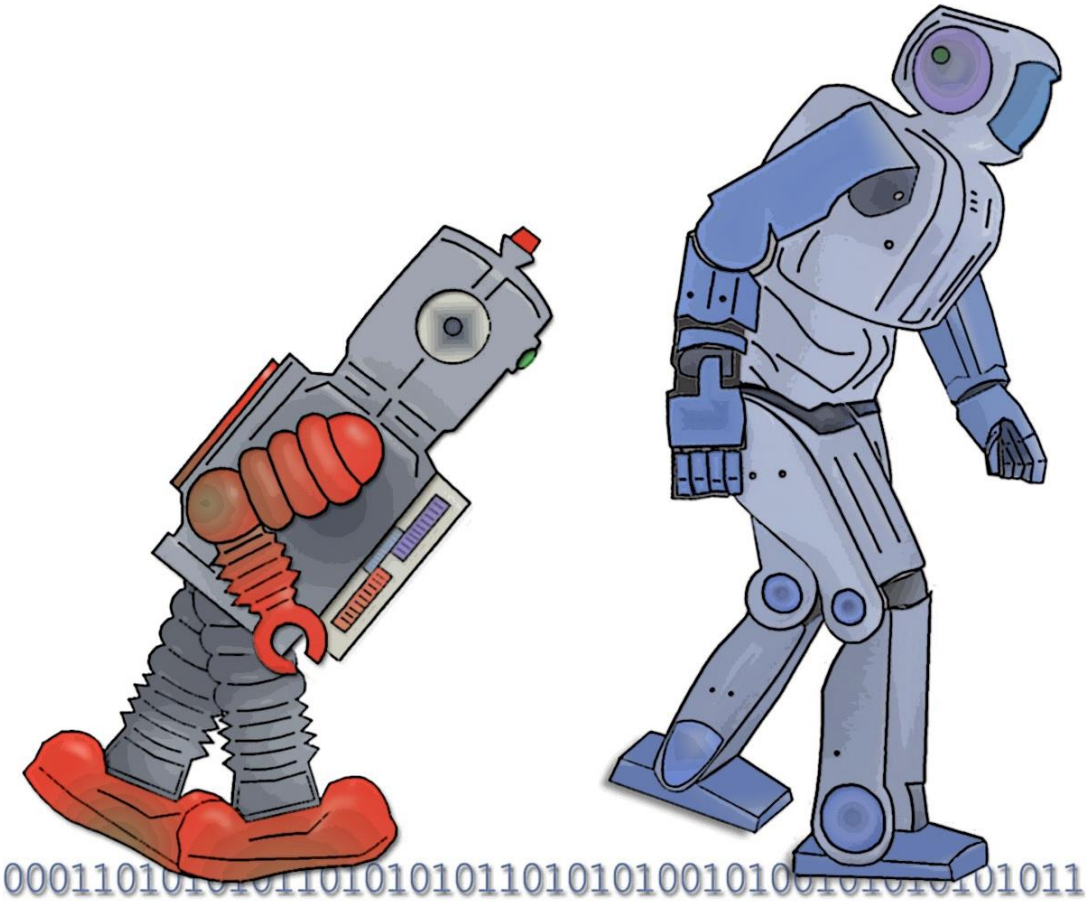
Test automation practice with Selenium Webdriver

Bauman Norbert
Csapó Péter
Mikó Szilárd

EPAM Systems, Budapest, 2017

Advanced Agenda

1. Interrogation
2. Manipulation
3. Window handling
4. Screenshots



CHAPTER 1

INTERROGATION

Interrogation

Window Title

- driver.getTitle()

Current URL

- driver.getCurrentUrl()

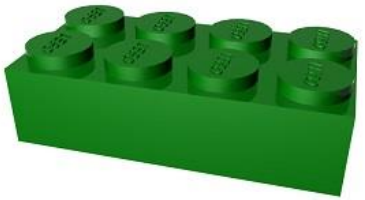
Page Source

- driver.getPageSource()

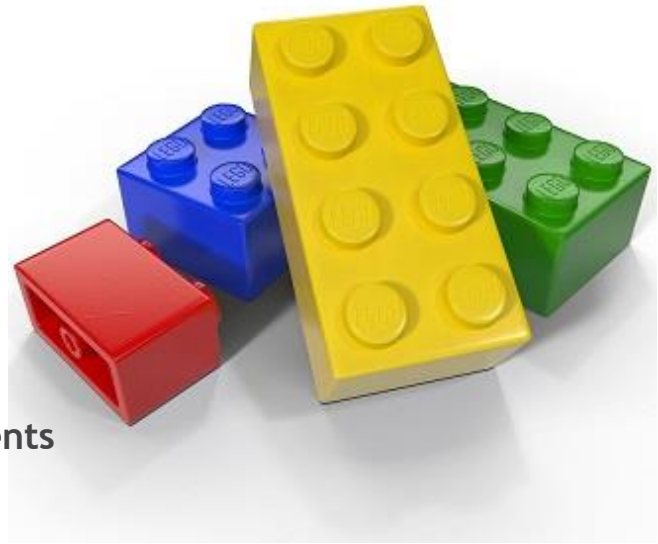
Interrogation

Locating web elements

- driver.findElement(org.openqa.selenium.By)
  - 0 match -> throws exception
  - 1 match -> returns a WebElement instance
  - 2+ matches -> returns only the first match from web page



- driver.findElements(org.openqa.selenium.By)
  - 0 match -> returns an empty list
  - 1 match -> returns a list with one WebElement
  - 2+ matches -> returns list with all matching WebElements



Interrogation

By class

- Supports various locator strategies
- By locating mechanisms
  - id
  - className
  - linkText
  - partialLinkText
  - name
  - tagName
  - cssSelector
  - xpath

Interrogation

Inspecting elements in web browsers

- Firefox
  - Firebug add-on (Right click -> Inspect element / F12)
  - Firefinder add-on (Try out your CSS & Xpath expressions)
- Chrome
  - Built-in (Right click -> Inspect element / F12)
- IE
  - Built-in (Tools -> Developer Tools / F12)

Interrogation

Id

- driver.findElement(By.id(”some_id”));
- Ideal solution, however…
  - Ids don’t always exist
  - Their uniqueness is not enforced
  - Used by developers as well

ClassName

- driver.findElement(By.className(”some_class_name”));

Interrogation

Linktext

- driver.findElement(By.linkText(”Sign in”));
- driver.findElement(By.partiallinkText(”Sign”));

Name

- <input id=”modCustLoginPassword” name=”password”>
- driver.findElement(By.name(”password”));

Interrogation

tagName

- <label>Email address</label>
- driver.findElement(By.tagName("label"));

Support classes

- Return all that matches each of the locators in sequence
  - driver.findElements(new ByChained(by1, by2))
- Return all that matches any of the locators in sequence
  - driver.findElements(new ByAll(by1, by2))

Interrogation

CssSelector

- Absolute path
  - driver.findElement(By.cssSelector(”html>body>div>p>input”));
- Relative path
  - driver.findElement(By.cssSelector(”input”));
- Attribute selection
  - driver.findElement(By.cssSelector(”button[name]”));
  - driver.findElement(By.cssSelector(”button[name=,cancel’]”));
  - driver.findElement(By.cssSelector(”img:not[alt]”));

Interrogation

CssSelector

- Id selection
  - driver.findElement(By.cssSelector(”#save”));
- Class selection
  - driver.findElement(By.cssSelector(”.login”));
- Combined selection
  - driver.findElement(By.cssSelector(”button#save”));
  - driver.findElement(By.cssSelector(”input.login”));

Interrogation

CssSelector

- First matching child of the specified tag
  - driver.findElement(By.cssSelector(”div#students:first-child”));
- Nth matching child of the specified tag
  - driver.findElement(By.cssSelector(”#loginForm:nth-child(3)”));
- First matching enabled tag
  - driver.findElement(By.cssSelector(”button:enabled”));

Interrogation

XPath

- Absolute path
  - driver.findElement(By.xpath(”html/body/p/input”));
- Relative path
  - driver.findElement(By.xpath(”//input”));
- Attribute selection
  - driver.findElement(By.xpath(”//input[@id=’username’]”));
  - driver.findElement(By.xpath(”//*[@id=’myId’]”))

Interrogation

Element interrogation

- element.getText();
- element.getAttribute();
- element.getTagName();
- element.isDisplayed();
- element.isEnabled();
- element.isSelected(); -> checkbox is selected or not
- selectElement.isMultiple(); -> multi select listbox or not
- selectElement.getOptions(); -> listbox select options

CHAPTER 2

MANIPULATION

Manipulation

Click

- element.click()
  - Button
  - Link
  - Checkbox
  - Combobox

Submit

- form.submit()
  - Form

Manipulation

Shift + Click

- New Actions(driver).keyDown(Keys.SHIFT).click(element).

  keyUp(Keys.SHIFT).build().perform();

Special Actions

- new Actions(driver).moveToElement(element).build().perform();
- new Actions(driver).contextClick().build().perform();
- new Actions(driver).doubleClick().build().perform();
- new Actions(driver).clickAndHold().build().perform();
- new Actions(driver).release().build().perform();

Manipulation

Type text

- element.sendKeys(”string”)
  - Input field

Clear text

- element.clear()

Manipulation

Listbox Selection

- new Select(element).selectByIndex(elementCount)

Listbox Manipulating Commands

- select[ByIndex, ByVisibleText, ByValue]
- deselect[ByIndex, ByVisibleText, ByValue]
- deselectAll()

CHAPTER 3

WINDOW HANDLING

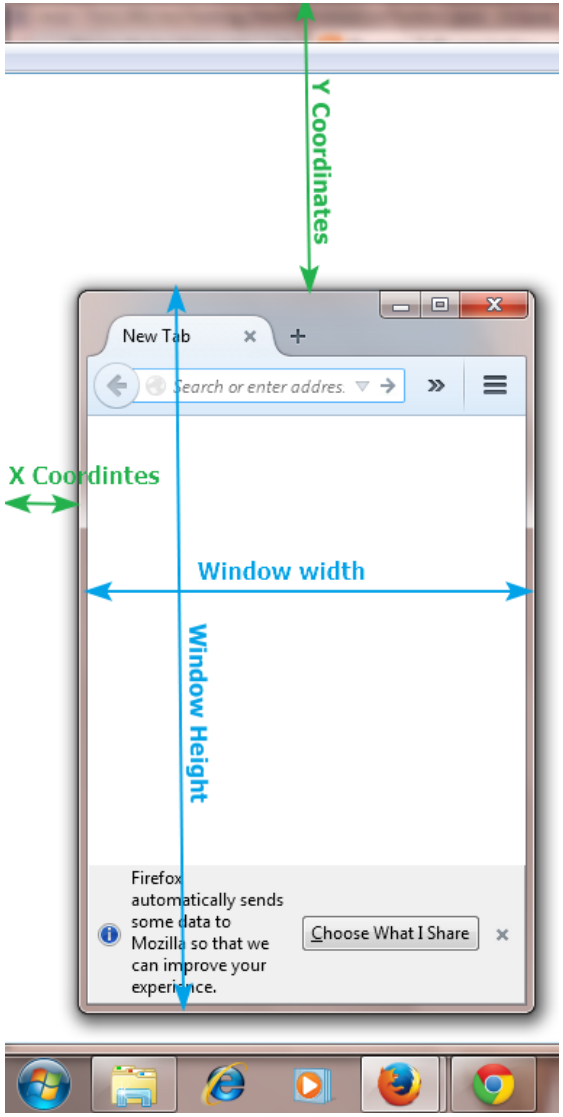
Window Handling

Size

- driver.manage().window().getSize() .getHeight(); .getWidth();
- driver.manage().window().setSize(Dimension d);
- driver.manage().window().maximize();

Position

- driver.manage().window().getPosition() .getX(); .getY();
- driver.manage().window().setPosition(Point p);



Window Handling

Handles

- String windowHandle = driver.getWindowHandle();
- Iterator<String> windowIterator = browser.getWindowHandles();

Switch To

- driver.switchTo().window(windowHandle);

CHAPTER 4

SCREENSHOTS

Screenshots

Advantages

- Keep track of changing UI
- Store pages with error

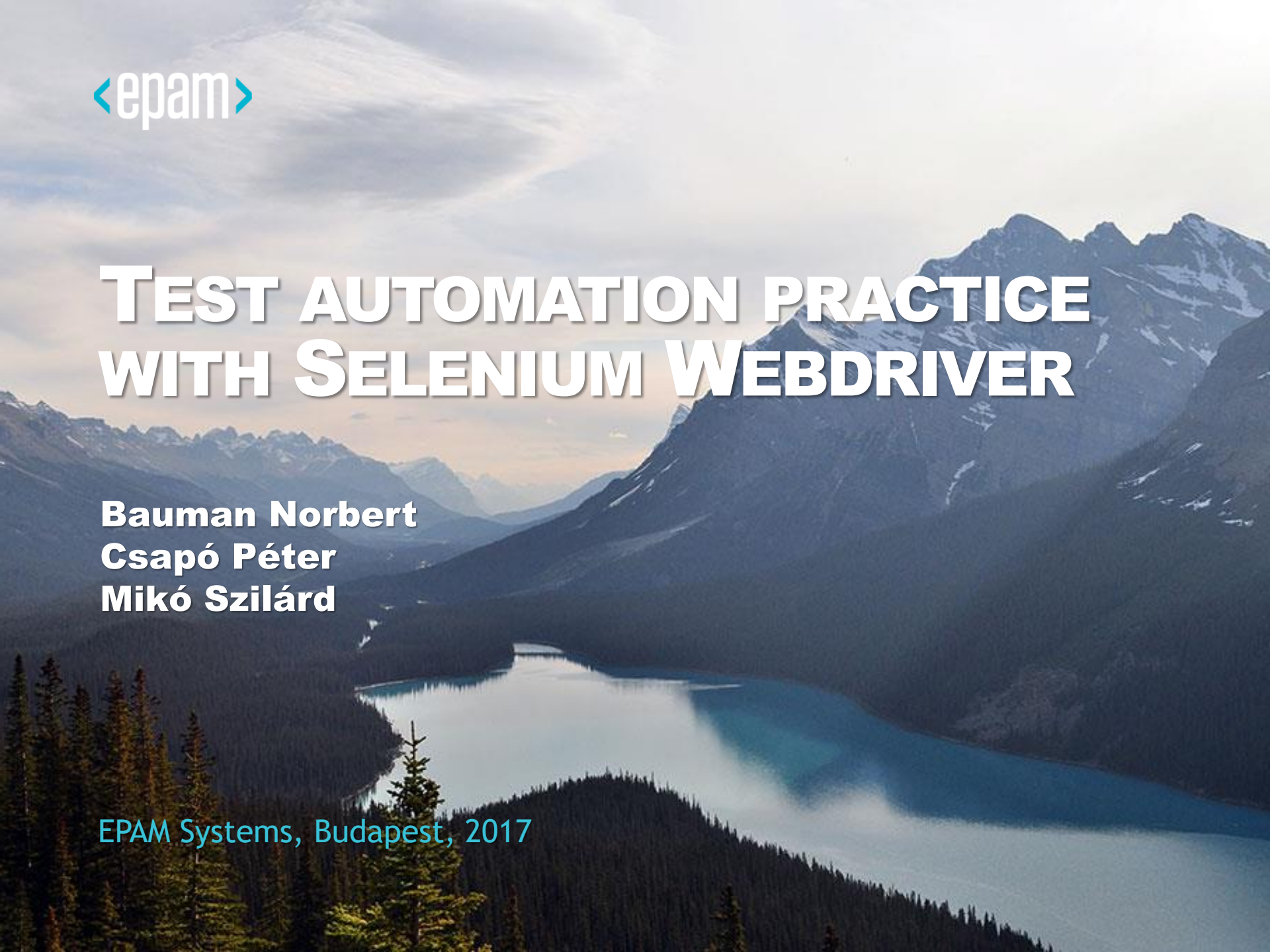
Example

- File screenshot =

```
((TakesScreenshot)driver).getScreenshotAs(OutputType.FILE);
```

- FileUtils.copyFile(screenshot, new File(fileSource));

Questions



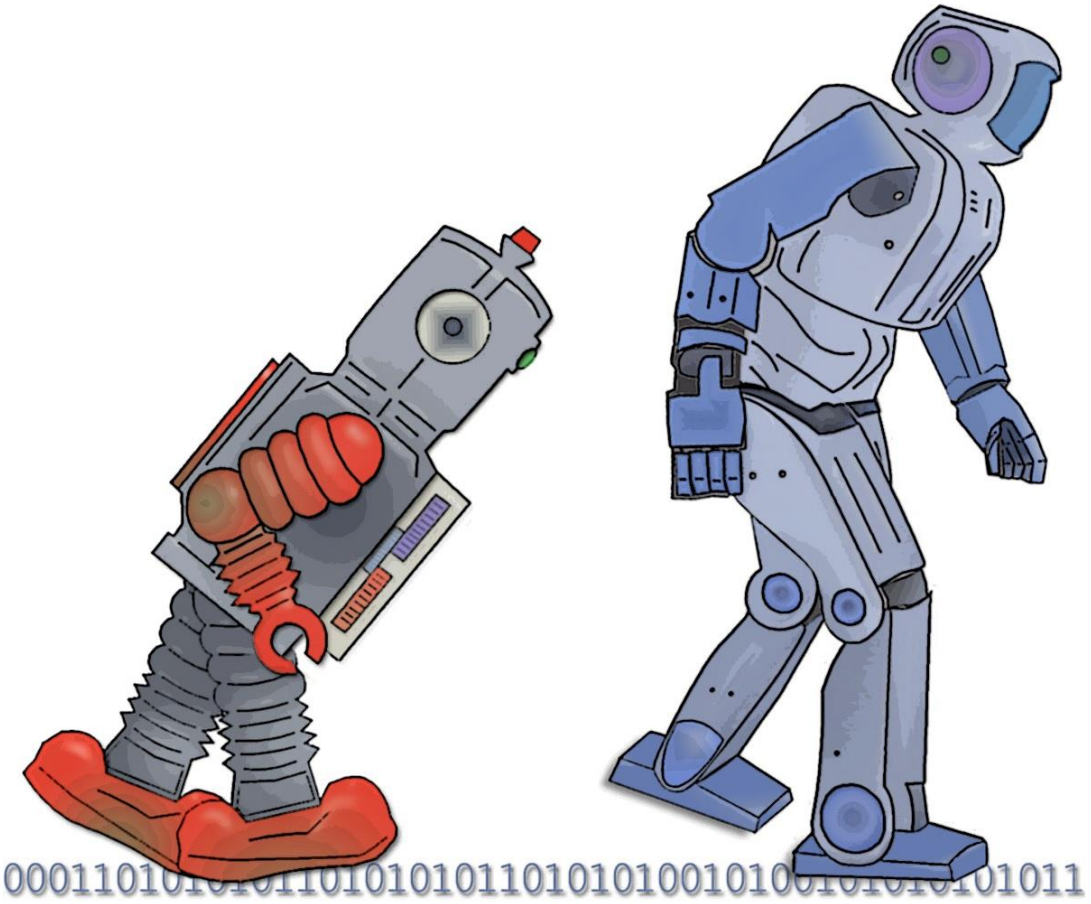
Test automation practice with Selenium Webdriver

Bauman Norbert
Csapó Péter
Mikó Szilárd

EPAM Systems, Budapest, 2017

Advanced Agenda

1. Synchronization
2. Cookies
3. Data Driven Testing



CHAPTER 1

SYNCHRONIZATION

Synchronization

Page Load Timeout

- Sets the amount of time to wait for a page load to complete
- Global setting of the Webdriver object
- Negative value means indefinite wait time

Example

- driver.manage().timeouts(). pageLoadTimeout(30, TimeUnit.SECONDS);

Synchronization

Implicit Wait

- Specifies the waiting time for element not immediately visible
- Global setting of the Webdriver object
- 0 by default

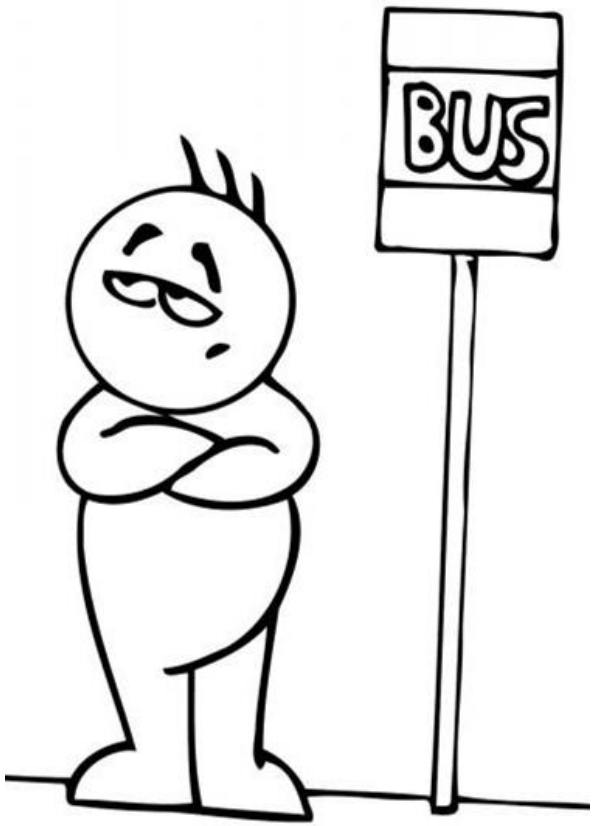
Example

- driver.manage().timeouts().implicitlyWait(5, TimeUnit.SECONDS);

Synchronization

Explicit Wait

- Waiting for a certain condition
- Poor alternative
  - Thread.sleep(1000);
- Recommended
  - WebDriverWait class



Example

- WebDriverWait wait = new WebDriverWait(driver, TIME_OUT_IN_SECONDS);
- wait.until(ExpectedConditions.method);

Synchronization

ExpectedConditions class

- presenceOfElementLocated(By locator)
- textToBePresentInElement(WebElement element, java.lang.String text)
- titleContains(java.lang.String title)
- visibilityOf(WebElement element)
- invisibilityOfElementLocated(By locator)
- elementToBeSelected(WebElement element)
- elementToBeClickable(By locator)

Click here for more ExpectedConditions

Synchronization

Fluent Wait

- Provides more control than ExplicitWait:
  - Specify polling time
  - Ignore exception classes

Example

```
Wait<WebDriver> wait = new FluentWait<WebDriver>(driver)
    .withTimeout(5, TimeUnit.SECONDS).pollingEvery(1, TimeUnit.SECONDS)
    .ignoring(NoSuchElementException.class);
wait.until(ExpectedConditions.alertIsPresent());
```

CHAPTER 2

COOKIES

Cookies

Introduction

- Useful for testing the login feature
- Getting or setting session IDs
- Cookie attributes
  - Name
  - Value
  - Domain
  - Path
  - Expiry
  - Secure
  - Http only

Cookies

Interrogation

- Get all cookies from the current session
  - driver.manage().getCookies();
- Get cookie with a given name
  - driver.manage().getCookieNamed(cookieToTest);

Cookies

Manipulation

- Delete all cookies from the current session
  - driver.manage().deleteAllCookies()
- Delete a specific cookie
  - driver.manage().deleteCookie(TestCookie);
- Delete cookie with a given name
  - driver.manage().deleteCookieNamed(cookieToTest);

Cookies

Manipulation

- Add a specific cookie
  - Cookie cookie = new Cookie("mycookie", "123456");
  - driver.manage().addCookie(cookie);
- Domain attribute is the current document by default